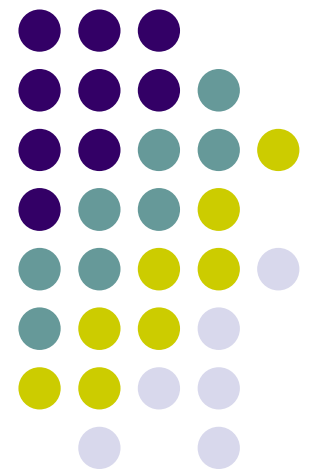


# Ingeniería de Software

---

## Diagramas de Secuencia



# Que son los Diagramas de Secuencia



- Un diagrama de secuencias muestra la interacción de un conjunto de objetos de una aplicación a través del tiempo, en el cual se indicaran los módulos o clases que formaran parte del programa y las llamadas que se hacen cada uno de ellos para realizar una tarea determinada, por esta razón permite observar la perspectiva cronológica de las interacciones.

# Elementos

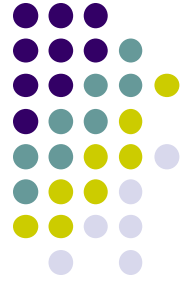
## Rol de la Clase



- El rol de la clase describe la manera en que un objeto se va a comportar en el contexto. No se listan los atributos del objeto.

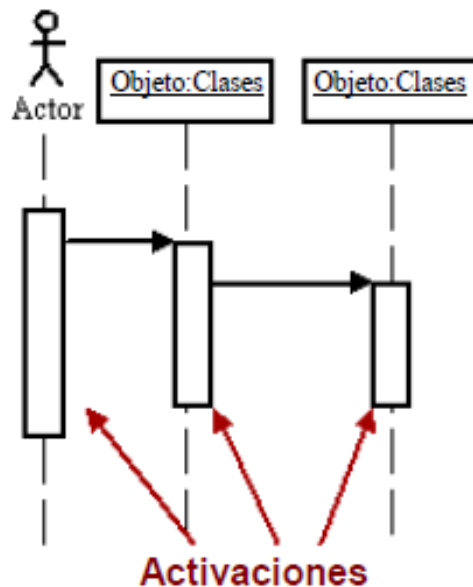
Objeto : Clase

Objeto de clase



# Activacion

- Los cuadros de activación representan el tiempo que un objeto necesita para completar una tarea.



Activación de una clase



# Mensajes

- Los mensajes son flechas que representan comunicaciones entre objetos. Las medias flechas representan mensajes asincrónicos. Los mensajes asincrónicos son enviados desde un objeto que no va a esperar una respuesta del receptor para continuar con sus tareas.



- Representados por una línea continua y una punta de flecha sólida. Este símbolo se utiliza cuando un remitente debe esperar una respuesta a un mensaje antes de proseguir. El diagrama debe mostrar el mensaje y la respuesta.



**Símbolo de mensaje sincrónico**

- Representados por una línea continua y una punta de flecha simple. Los mensajes asíncronos son aquellos que no necesitan una respuesta para que el remitente siga adelante. Solo la llamada se debe incluir en el diagrama.

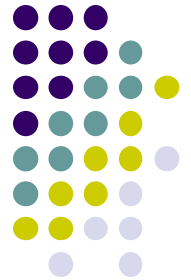


**Símbolo de mensaje asíncrono**

- Representados por una línea discontinua y una punta de flecha simple.



**Símbolo de mensaje de respuesta  
asíncrono**



- Representados por una línea discontinua y una punta de flecha simple. Estos mensajes se envían a las líneas de vida para crearse por sí solos.



- Están representados con una línea discontinua y una punta de flecha simple. Estos mensajes son las respuestas a las llamadas.



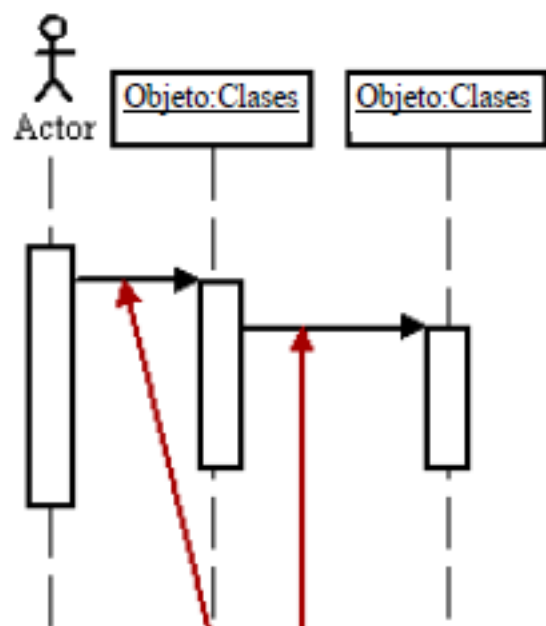


- Están representados por una línea continua y una punta de flecha sólida, seguida de un símbolo X. Estos mensajes indican la destrucción de un objeto y están ubicados en su ruta de la línea de vida.








**Símbolo de eliminar mensaje**





**Mensajes**

Líneas que  
se pueden  
utilizar para  
representar  
los  
mensajes

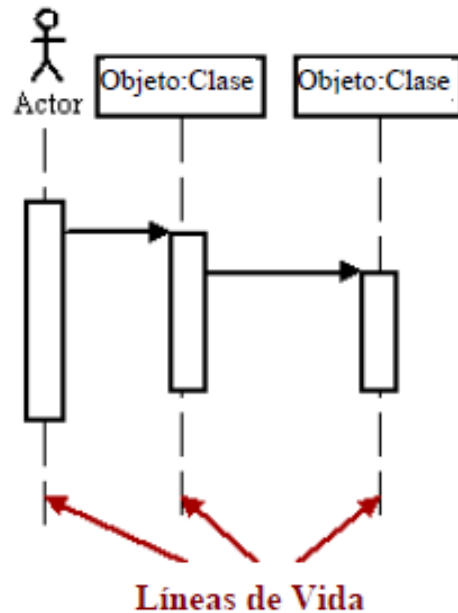
Flecha	Tipo de mensaje
	Simple
	Sincrónico
	Asincrónico
	Rechazado
	Time out

Mensajes



# Líneas de Vida

- Las líneas de vida son verticales y en línea de puntos, ellas indican la presencia del objeto durante el tiempo.

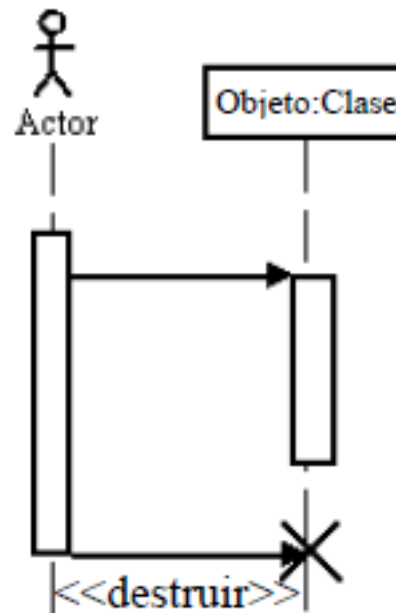


Línea de Vida



# Destruccion de Objetos

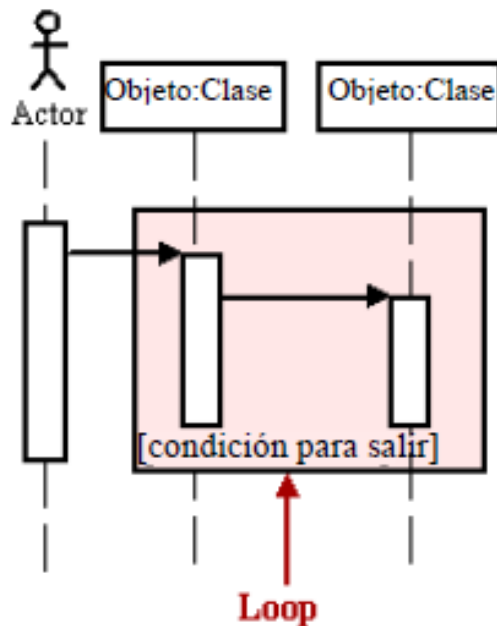
- Los objetos pueden ser eliminados tempranamente usando una flecha etiquetada “<<destruir>>” que apunta a una X.



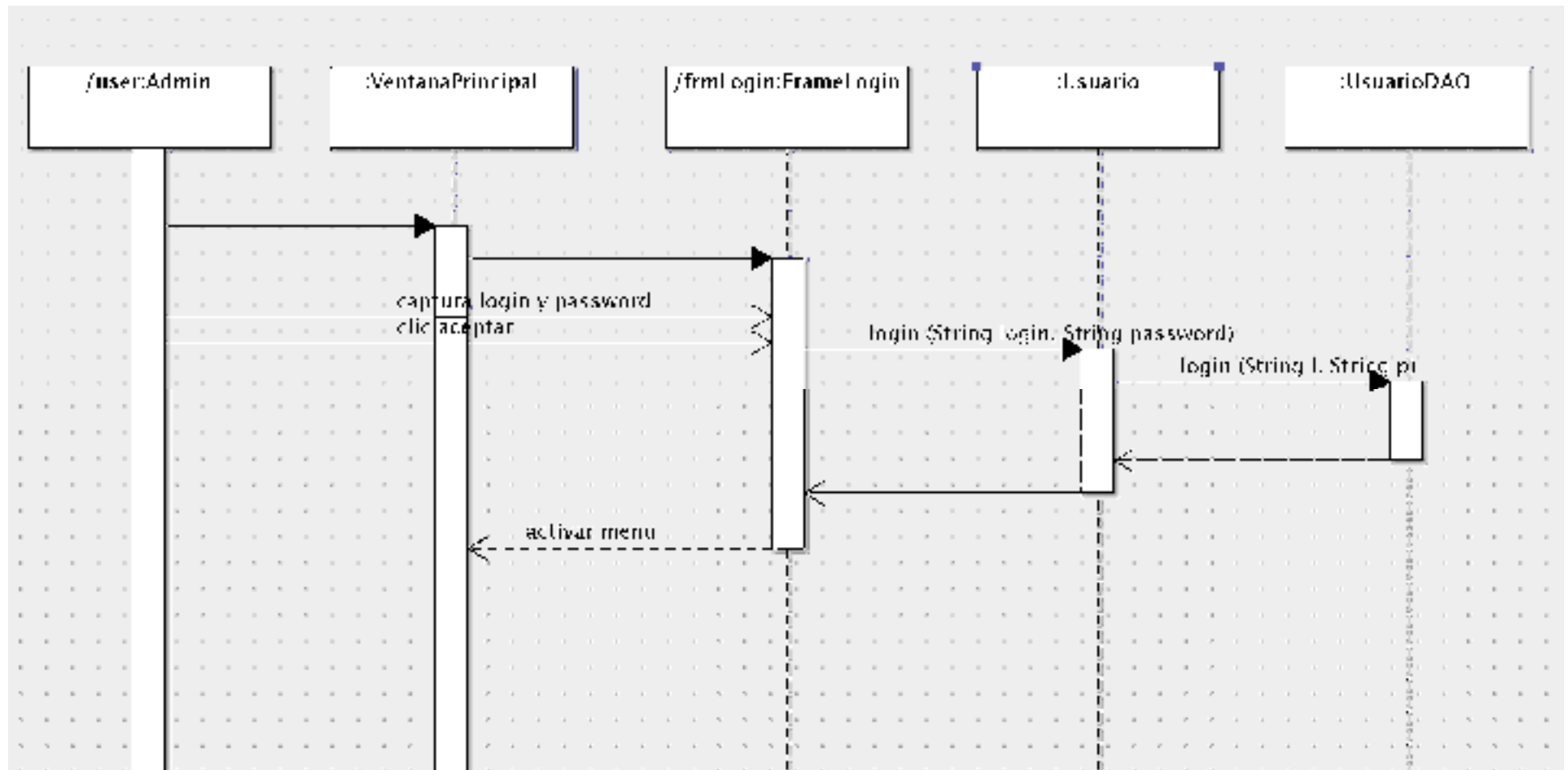
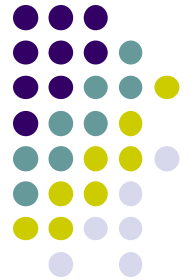
# Loops



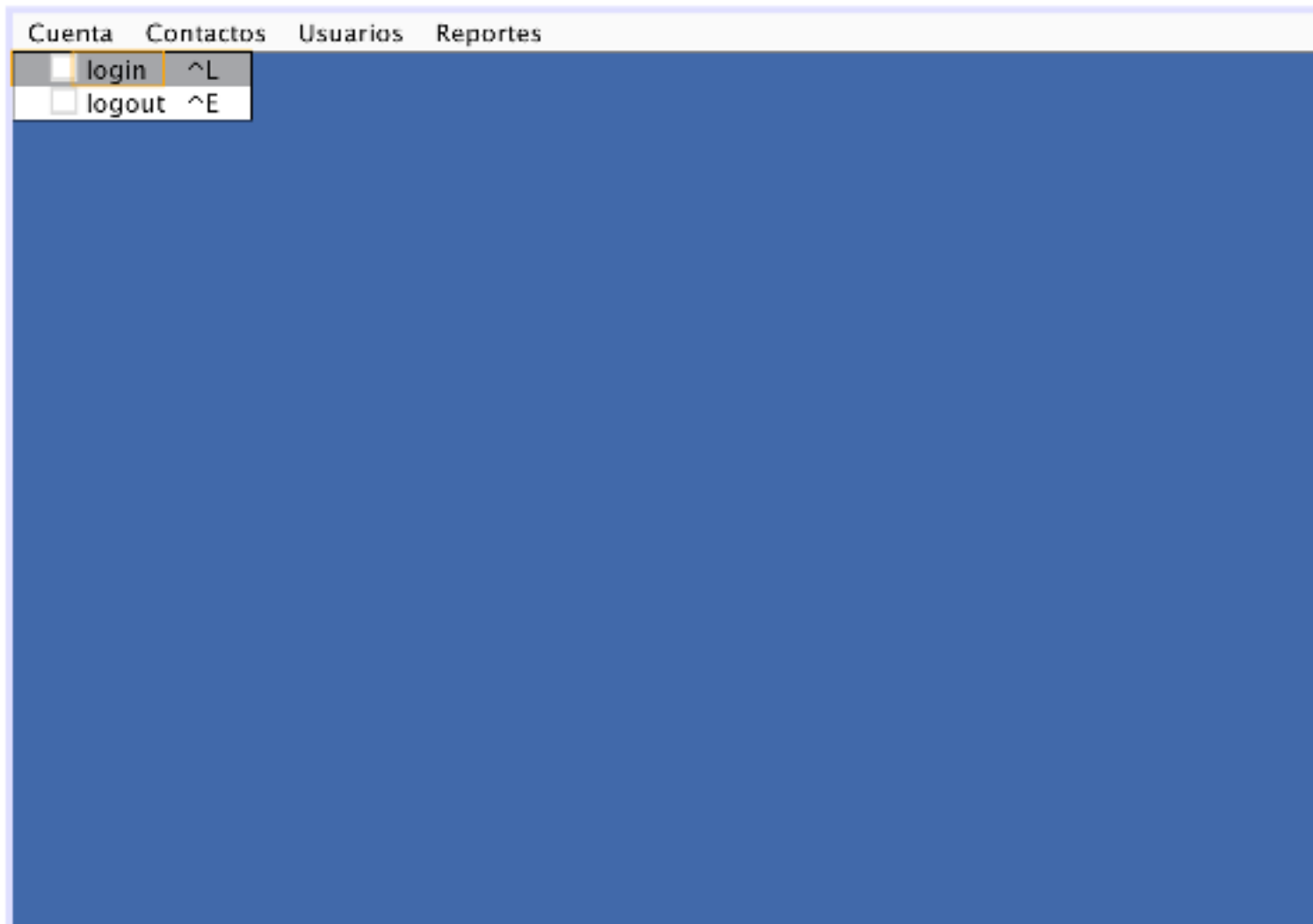
- Una repetición o loop en un diagrama de secuencias, es representado como un rectángulo. La condición para abandonar el loop se coloca en la parte inferior entre corchetes [ ].



# Ejemplo: Login



# VentanaPrincipal.java



# VentanaPrincipal.java

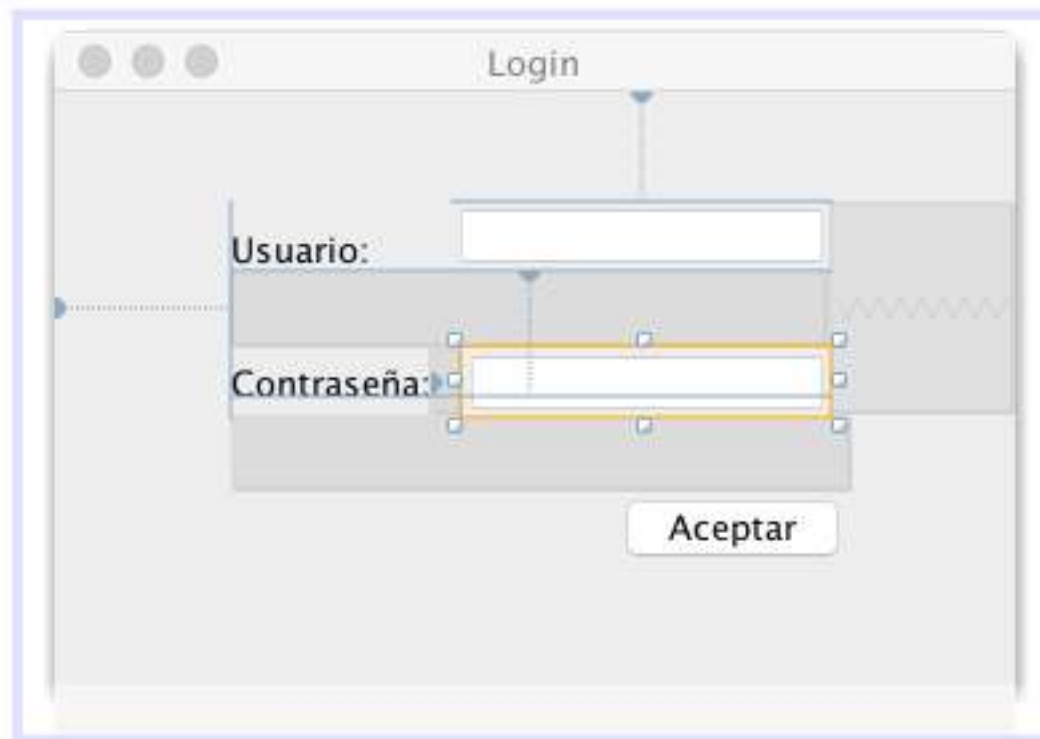


Generated Code

```
private void loginActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    FrameLogin frmLogin=new FrameLogin();  
    this.desktopPane.add(frmLogin);  
    frmLogin.setVisible(true);  
    this.repaint();  
}
```

```
/**  
 * @param args the command line arguments  
 */  
public static void main(String args[]) {
```

# FrameLogin.java



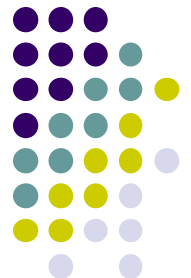


# FrameLogin.java



```
private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {  
    // TODO add your handling code here:  
    if (validarCampos()) {  
        try {  
            if (udao.login(login, password)) {  
                JOptionPane.showConfirmDialog(this, "Login exitoso!!");  
            }  
        } catch (SQLException ex) {  
            JOptionPane.showMessageDialog(this, "Error de conexion de BD");  
        }  
    }  
}
```

```
private boolean validarCampos() {  
    boolean resultado = false;  
  
    if ((jTextField1.getText().equals("")) && (jPasswordField1.getPassword().length == 0)) {  
        JOptionPane.showMessageDialog(this, "Campos vacios, revisar los datos y volver a intentar",  
        JOptionPane.ERROR_MESSAGE);  
    } else {  
        login = jTextField1.getText();  
        password = new String(jPasswordField1.getPassword());  
        resultado = true;  
    }  
  
    return resultado;  
}
```



```
*/
public class UsuarioDAO implements InterfazDAO {

    private ConexionDataBase db;

    public UsuarioDAO() {...3 lines }
    public boolean insertar(Object obj) throws SQLException {...7 lines }
    public boolean update(Object obj) throws SQLException {...4 lines }
    public boolean delete(String id) throws SQLException {...4 lines }
    public ArrayList obtenerTodo() throws SQLException {...5 lines }
    public Object buscarPorId(String id) throws SQLException {...5 lines }

    public boolean login(String l, String p) throws SQLException{
        boolean res=false;
        try {
            String query= "SELECT * FROM usuarios where login=? and password=?";
            PreparedStatement pstmt=db.getConnection().prepareStatement(query);
            pstmt.setString(1, l);
            pstmt.setString(2, p);

            ResultSet rst= pstmt.executeQuery();

            if (rst.next()) {
                res=true;
            }

        } catch (SQLException ex) {
            throw ex;
        }
        return res;
    }
}
```